D4.8 Annual Report on Quantum-Safe Functional Encryption schemes Y2

Document Identification			
Status	Final	**Due Date**	31/12/2019
Version	1.0	**Submission Date**	30/12/2019

Related WP	WP4	**Document Reference**	D4.8
Related Deliverable(s)	D4.1,D4.2,D4.7,D4.10	**Dissemination Level(*)**	PU
Lead Participant	ENS	**Lead Author**	Michel Abdalla
Contributors	ENS,UEDIN	**Reviewers**	Ward Beullens (KU Leuven)

Keywords:
Functional Encryption Schemes, Quantum-Safe

This document is issued within the framework and for the purpose of the FENTEC project. This project has received funding from the European Union's Horizon2020 under Grant Agreement No. 780108. The opinions expressed and arguments employed herein do not necessarily reflect the official views of the European Commission.

(*) Dissemination level.-PU: Public, fully open, e.g. web; CO: Confidential, restricted under conditions set out in Model Grant Agreement; CI: Classified, Int = Internal Working Document, information as referred to in Commission Decision 2001/844/EC.

Document Information

List of Contributors	
Name	Partner
Michel Abdalla	ENS
Hendrik Waldner	UEDIN

Document History			
Version	Date	Change editors	Changes
0.1	18/12/2019	Hendrik Waldner (UEDIN)	ToC
0.2	19/12/2019	Hendrik Waldner (UEDIN)	Created first full version
0.3	20/12/2019	Michel Abdalla (ENS)	Version for internal review
0.4	23/12/2019	Michel Abdalla (ENS)	Addressed reviewers comments
1	30/12/2019	Michel Abdalla (ENS)	Final version

Quality Control		
Role	Who (Partner short name)	Approval Date
Deliverable Leader	Michel Abdalla (ENS)	30/12/2018
Technical Manager	Michel Abdalla (ENS)	30/12/2018
Quality Manager	Diego Esteban (ATOS)	30/12/2018
Project Coordinator	Francisco Gala (ATOS)	30/12/2018

Table of Contents

List of Figures

List of Acronyms

Acronym	Description
DCR	Decisional Composite Residuosity
DDH	Decisional Diffie-Hellman
DMCFE	Decentralized Multi-Client Functional Encryption
FE	Functional Encryption
LWE	Learning With Errors
MCFE	Multi-Client Functional Encryption
MDDH	Matrix Decisional Diffie-Hellman
MIFE	Multi-Input Functional Encryption
PPT	Probabilistic Polynomial Time
WP	Work Package

Executive Summary

Most of the existing functional encryption schemes in use today are based on the presumed hardness of the discrete-log and the integer-factorization problems, which are known to be insecure with respect to quantum computers [14]. To prevent the collapse of the cryptographic protocols relying on these schemes, it is important to develop alternative solutions based on mathematical problems that are unrelated to factoring and discrete log and that may be impervious to attacks by quantum computers. Hence, one of the main goals of WP4 is to design quantum-safe functional encryption alternatives that use lattices as their source of computational hardness. In this deliverable, we build up on the results of Deliverable 4.7 and describe the progress made within the FENTEC project towards this goal.

More precisely, we solve an open problem stated in the Deliverables 4.1 and 4.7 which considers the central authority responsible for the generation of the functional keys. More precisely, we present an information-theoretically secure compiler constructed by Abdalla et al. in [2] that allows to generate functional keys in a decentralized manner. Next, we also extend the analysis given in Deliverable 4.7 to show that a natural variant of the multi-input functional encryption (MIFE) scheme for the inner-product functionality by Abdalla et al. in [5] also fulfills the stronger notion of multi-client functional encryption (MCFE), as shown in [2]. Finally, we recap the quantum-safe schemes that can be used to instantiate the new MCFE variant.

1 Introduction

Functional encryption (FE) [7, 12] is a generalization of the notion of public-key encryption, which allows fine-grained access control over encrypted data. Besides the classical encryption and decryption procedures, functional encryption schemes consists of a key derivation algorithm, which allows the owner of a master secret key to derive keys with more restricted capabilities. These derived keys sk_f are called functional decryption keys and are associated with a function f. Using the key sk_f for the decryption of a ciphertext $\mathsf{Enc}(x)$ generates the output $f(x)$. During this decryption procedure no more information is revealed about the underlying plaintext than $f(x)$.

In the case of classical functional encryption, the functional decryption procedure takes as input a single ciphertext $\mathsf{Enc}(x)$. A natural extension is the multi-input setting, where the decryption procedure takes as input n different ciphertexts and outputs a function applied on the n corresponding plaintexts. Such a scheme is called multi-input functional encryption (MIFE) scheme [11]. In a MIFE scheme, each ciphertext can be generated independently.

An important use case of MIFE considers multiple parties or clients, where each party P_i generates a single ciphertext of the tuple. The ciphertext generated by party P_i is often said to correspond to *position* or *slot* i. In the multi-client setting, it becomes natural to assume that each party has a different secret/encryption key sk_i that can be corrupted by the adversary. We call such a scheme a multi-client functional encryption (MCFE) scheme [9, 11].

While the exact terminology varies in the literature, as in [2], we assume here that a MCFE scheme is always supposed to be secure against corruption of the parties encrypting messages. In a MIFE scheme, on the other hand, all the parties may use the same encryption key and there is no security against corruption.

1.1 Purpose of the Document

The goal of this deliverable is to describe our contributions to the design of practical quantum-safe functional encryption schemes within the FENTEC project. Towards this goal, we present two contributions.

Our first contribution is to address one of the open problems listed in Deliverables 4.1 and 4.7, concerning removing the need for a central authority during the generation of the functional keys. More precisely, we present a compiler developed by Abdalla et al. in [2] in the context of FENTEC which allows us to generate functional keys in a simple decentralized manner. The solution supports encryption labels and can be applied to any existing multi-input functional encryption (MIFE) scheme satisfying a structural property called special key-derivation. As the compiler does not rely on any computational assumption, it therefore preserves the quantum safety of the underlying scheme.

Our second contribution is to prove security against adaptive corruptions for a MCFE variant of the MIFE scheme by Abdalla, Catalano, Fiore, Gay, and Ursu [5], in which their unique encryption and secret key is split into individual secret keys for each party a natural way. Like the original construction in [5], the new MCFE variant is generic, in the sense that it can transform any single-input FE that satisfies some structural properties into a MCFE, under the same assumption.

Finally, in order to instantiate the new MCFE scheme, we recall the description of two FE schemes for the inner-product functionality by Agrawal et al. [6] and by Abdalla et al. [4]. Since both schemes are based on the Learning-With-Errors (LWE) problem, the resulting schemes are the

first MCFE schemes for the inner-product functionality (without labels) based on quantum-safe assumptions.

1.2 Structure and Methodology

Section 2 first recalls some of the definitions and basic tools that are used in the remainder of the document, such as notations, complexity assumptions, and security definitions for MCFE. Section 3 then describes the decentralization compiler by Abdalla et al. [2], developed in the context of the FENTEC project, which allows for the decentralization of MCFE schemes satisfying an additional structure property, called special key derivation. Next, Section 4 describes the new generic MCFE construction from a single-input functional inner-product encryption based on the MIFE scheme by Abdalla et al. [5]. Section 5 describes two concrete quantum-safe single-input FE schemes that be used to instantiate the generic construction in Section 4, one by Agrawal et al. [6] and one by Abdalla et al. [4]. Finally, Section 6 concludes by recalling our main contributions.

2 Basic tools

In this section, we recall some of the definitions and basic tools, that will be used in the remainder of the document. This contains the LWE assumption and the definitions of different types of multi-client functional encryption schemes.

2.1 Notation and conventions

We denote with $\lambda \in \mathbb{N}$ a security parameter. A *probabilistic polynomial time* (PPT) algorithm $\mathcal{A}$ is a randomized algorithm for which there exists a polynomial $p(\cdot)$ such that for every input x the running time of $\mathcal{A}(x)$ is bounded by $p(|x|)$. We say that a function $\varepsilon : \mathbb{N} \to \mathbb{R}^+$ is *negligible* if for every positive polynomial $p(\lambda)$ there exists $\lambda_0 \in \mathbb{N}$ such that for all $\lambda > \lambda_0$: $\varepsilon(\lambda) < 1/p(\lambda)$. If S is a set, $x \xleftarrow{R} S$ denotes the process of selecting x uniformly at random in S. If $\mathcal{A}$ is a probabilistic algorithm, $y \xleftarrow{R} \mathcal{A}(\cdot)$ denotes the process of running $\mathcal{A}$ on some appropriate input and assigning its output to y. For a positive integer n, we denote by $[n]$ the set $\{1, \ldots, n\}$. We denote vectors $\mathbf{x} = (x_i)$ and matrices $\mathbf{A} = (a_{i,j})$ in bold. For a set S (resp. vector $\mathbf{x}$) $|S|$ (resp. $|\mathbf{x}|$) denotes its cardinality (resp. number of entries). Also, given two vectors $\mathbf{x}$ and $\mathbf{x}'$ we denote by $\mathbf{x}\|\mathbf{x}'$ their concatenation. By $\equiv$, we denote the equality of statistical distributions, and for any $\varepsilon > 0$, we denote by $\approx_\varepsilon$ the ε-statistical difference of two distributions.

2.2 Learning With Errors (LWE)

Since this report only considers quantum-safe schemes, we now recall the *Learning-With-Errors* (LWE) complexity assumption used in some of these schemes.

Definition 1 (Learning With Errors (LWE) assumption) *Let q, α, m be functions of a parameter n. For a secret $\mathbf{s} \in \mathbb{Z}_q^n$, the distribution $A_{q,\alpha,s}$ over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ is obtained by sampling $\mathbf{a} \xleftarrow{R} \mathbb{Z}_q^n$ and an error $e \xleftarrow{R} \psi_{\mathbb{Z},\alpha,q}$ from an error distribution $\psi_{\mathbb{Z},\alpha,q}$, and returning $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e) \in \mathbb{Z}_q^{n+1}$. Let $U(\mathbb{Z}_q^{m\times(n+1)})$ denote the uniform distribution over $\mathbb{Z}_q^{m\times(n+1)}$. The Learning With Errors problem $\mathsf{LWE}_{q,\alpha,m}$ is as follows: For $\mathbf{s} \xleftarrow{R} \mathbb{Z}_q^n$, the goal is to distinguish between the distributions:*

$$\mathsf{D}_0(\mathbf{s}) := U(\mathbb{Z}_q^{m\times(n+1)}) \text{ and } \mathsf{D}_1(\mathbf{s}) := (A_{q,\alpha,\mathbf{s}})^m.$$

We say that a PPT algorithm $\mathcal{A}$ solves the $\mathsf{LWE}_{q,\alpha,m}$ problem if it distinguishes $\mathsf{D}_0(\mathbf{s})$ and $\mathsf{D}_1(\mathbf{s})$ (with non-negligible advantage over the random coins of $\mathcal{A}$ and the randomness of the samples) with non-negligible probability over the randomness of $\mathbf{s}$. The LWE assumption states that no such adversary exists.

After introducing the computational hardness assumption on which we rely in this deliverable, we introduce the notion of multi-client functional encryption (MCFE).

Definition 2 ***(Multi-Client Functional Encryption)*** *Let $\mathcal{F} = \{\mathcal{F}_\rho\}_\rho$ be a family (indexed by ρ) of sets $\mathcal{F}_\rho$ of functions $f : \mathcal{X}_{\rho,1} \times \cdots \times \mathcal{X}_{\rho,n_\rho} \to \mathcal{Y}_\rho$.[1] Let $\mathsf{Labels} = \{0,1\}^*$ or $\{\perp\}$ be a set of labels. A multi-client functional encryption scheme (MCFE) for the function family $\mathcal{F}$ and the label set Labels is a tuple of five algorithms $\mathsf{MCFE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$:*

[1] All the functions inside the same set $\mathcal{F}_\rho$ have the same domain and the same range.

$\mathsf{Setup}(1^\lambda, 1^n)$**:** *Takes as input a security parameter* λ *and the number of parties* n*, and generates public parameters* pp*. The public parameters implicitly define an index* ρ *corresponding to a set* $\mathcal{F}_\rho$ *of* n*-ary functions (i.e.,* $n = n_\rho$*).*

$\mathsf{KeyGen}(\mathsf{pp})$**:** *Takes as input the public parameters* pp *and outputs* n *secret keys* $\{\mathsf{sk}_i\}_{i\in[n]}$ *and a master secret key* msk*.*

$\mathsf{KeyDer}(\mathsf{pp}, \mathsf{msk}, f)$**:** *Takes as input the public parameters* pp*, the master secret key* msk *and a function* $f \in \mathcal{F}_\rho$*, and outputs a functional decryption key* sk_f*.*

$\mathsf{Enc}(\mathsf{pp}, \mathsf{sk}_i, x_i, \ell)$**:** *Takes as input the public parameters* pp*, a secret key* sk_i*, a message* $x_i \in \mathcal{X}_{\rho,i}$ *to encrypt, a label* $\ell \in \mathsf{Labels}$*, and outputs ciphertext* $\mathsf{ct}_{i,\ell}$*.*

$\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}_f, \mathsf{ct}_{1,\ell}, \ldots, \mathsf{ct}_{n,\ell})$**:** *Takes as input the public parameters* pp*, a functional key* sk_f *and* n *ciphertexts under the same label* ℓ *and outputs a value* $y \in \mathcal{Y}_\rho$*.*

A scheme MCFE *is correct, if for all* $\lambda, n \in \mathbb{N}$*,* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$*,* $f \in \mathcal{F}_\rho$*,* $\ell \in \mathsf{Labels}$*,* $x_i \in \mathcal{X}_{\rho,i}$*, when* $(\{\mathsf{sk}_i\}_{i\in[n]}, \mathsf{msk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$ *and* $\mathsf{sk}_f \leftarrow \mathsf{KeyDer}(\mathsf{pp}, \mathsf{msk}, f)$*, we have*

$$\Pr\left[\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}_f, \mathsf{Enc}(\mathsf{pp}, \mathsf{sk}_1, x_1, \ell), \ldots, \mathsf{Enc}(\mathsf{pp}, \mathsf{sk}_n, x_n, \ell)) = f(x_1, \ldots, x_n)\right] = 1 \ .$$

When ρ is clear from context, the index ρ is omitted. When $\mathsf{Labels} = \{0,1\}^*$, we say that the scheme is *labeled* or *with labels*. When $\mathsf{Labels} = \{\bot\}$, we say that the scheme is *without labels*, and we often omit ℓ.

As noted in [9, 11], the security model of multi-client functional encryption is similar to the security model of standard multi-input functional encryption, as introduced in D4.2, except that instead of a single master secret key msk for encryption, each slot i has a different secret key sk_i and the keys sk_i can be individually corrupted. In addition, one also needs to consider corruptions to handle possible collusions between different parties. In the following, we define security as adaptive left-or-right indistinguishability under both static (sta), and adaptive (adt) corruption. We also consider three variants of these notions (one, any, pos) related to the number of encryption queries asked by the adversary for each slot.

Definition 3 ***(Security of MCFE)*** *Let* MCFE *be an MCFE scheme,* $\mathcal{F} = \{\mathcal{F}_\rho\}_\rho$ *a function family indexed by* ρ *and* Labels *a label set. For* $\mathrm{xx} \in \{\mathrm{sta}, \mathrm{adt}\}$*,* $\mathrm{yy} \in \{\mathrm{one}, \mathrm{any}, \mathrm{pos}\}$*, and* $\beta \in \{0,1\}$*, we define the experiment* xx-yy-$IND^{\mathsf{MCFE}}_\beta$ *in Fig. 1, where the oracles are defined as:*

Corruption oracle $\mathsf{QCor}(i)$**:** *Outputs the encryption key* sk_i *of slot* i*. We denote by* $\mathcal{CS}$ *the set of corrupted slots at the end of the experiment.*

Encryption oracle $\mathsf{QEnc}(i, x_i^0, x_i^1, \ell)$**:** *Outputs* $\mathsf{ct}_{i,\ell} = \mathsf{Enc}(\mathsf{pp}, \mathsf{sk}_i, x_i^\beta, \ell)$ *on a query* (i, x_i^0, x_i^1, ℓ)*. We denote by* $Q_{i,\ell}$ *the number of queries of the form* $\mathsf{QEnc}(i, \cdot, \cdot, \ell)$*.*

Key derivation oracle $\mathsf{QKeyD}(f)$**:** *Outputs* $\mathsf{sk}_f = \mathsf{KeyDer}(\mathsf{pp}, \mathsf{msk}, f)$*.*

and where Condition (*) *holds if all the following conditions hold:*

- *If* $i \in \mathcal{CS}$ *(i.e., slot* i *is corrupted): for any query* $\mathsf{QEnc}(i, x_i^0, x_i^1, \ell)$*,* $x_i^0 = x_i^1$*.*

$\textbf{sta-yy-IND}_\beta^{\mathsf{MCFE}}(\lambda, n, \mathcal{A})$	$\textbf{adt-yy-IND}_\beta^{\mathsf{MCFE}}(\lambda, n, \mathcal{A})$
$\mathcal{CS} \leftarrow \mathcal{A}(1^\lambda, 1^n)$	$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$
$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$	$(\{\mathsf{sk}_i\}_{i\in[n]}, \mathsf{msk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$
$(\{\mathsf{sk}_i\}_{i\in[n]}, \mathsf{msk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$	$\alpha \leftarrow \mathcal{A}^{\mathsf{QCor}(\cdot),\mathsf{QEnc}(\cdot,\cdot,\cdot,\cdot),\mathsf{QKeyD}(\cdot)}(\mathsf{pp})$
$\alpha \leftarrow \mathcal{A}^{\mathsf{QEnc}(\cdot,\cdot,\cdot,\cdot),\mathsf{QKeyD}(\cdot)}(\mathsf{pp}, \{\mathsf{sk}_i\}_{i\in\mathcal{CS}})$	**Output:** α if Condition (*) is satisfied, or a uniform bit otherwise
Output: α if Condition (*) is satisfied, or a uniform bit otherwise	

Figure 1: Security games for MCFE

- *For any label* $\ell \in \mathsf{Labels}$*, for any family of queries* $\{\mathsf{QEnc}(i, x_i^0, x_i^1, \ell)\}_{i\in[n]\setminus\mathcal{CS}}$*, for any family of inputs* $\{x_i \in \mathcal{X}_{\rho,i}\}_{i\in\mathcal{CS}}$*, for any query* $\mathsf{QKeyD}(f)$*, we define* $x_i^0 = x_i^1 = x_i$ *for any slot* $i \in \mathcal{CS}$*,* $\mathbf{x}^b = (x_1^b, \ldots, x_n^b)$ *for* $b \in \{0,1\}$*, and we require that:*

$$f(\mathbf{x}^0) = f(\mathbf{x}^1) \ .$$

  We insist that if one index $i \notin \mathcal{CS}$ *is not queried for the label* ℓ*, there is no restriction.*

- *When* $\mathrm{yy} = \mathrm{one}$*: for any slot* $i \in [n]$ *and* $\ell \in \mathsf{Labels}$*,* $Q_{i,\ell} \in \{0,1\}$*, and if* $Q_{i,\ell} = 1$*, then for any slot* $j \in [n] \setminus \mathcal{CS}$*,* $Q_{j,\ell} = 1$*. In other words, for any label, either the adversary makes no encryption query or makes exactly one encryption query for each* $i \in [n] \setminus \mathcal{CS}$*.*

- *When* $\mathrm{yy} = \mathrm{pos}$*: for any slot* $i \in [n]$ *and* $\ell \in \mathsf{Labels}$*, if* $Q_{i,\ell} > 0$*, then for any slot* $j \in [n]\setminus\mathcal{CS}$*,* $Q_{j,\ell} > 0$*. In other words, for any label, either the adversary makes no encryption query or makes at least one encryption query for each slot* $i \in [n] \setminus \mathcal{CS}$*.*

We define the advantage of an adversary $\mathcal{A}$ *in the following way:*

$$\mathsf{Adv}^{\text{xx-yy-}IND}_{\mathsf{MCFE},\mathcal{A}}(\lambda, n) = \big|\Pr[\text{xx-yy-IND}_0^{\mathsf{MCFE}}(\lambda, n, \mathcal{A}) = 1] - \Pr[\text{xx-yy-IND}_1^{\mathsf{MCFE}}(\lambda, n, \mathcal{A}) = 1]\big| \ .$$

A multi-client functional encryption scheme MCFE *is* xx-yy-*IND secure, if for any* n*, for any polynomial-time adversary* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that* $\mathsf{Adv}^{\text{xx-yy-}IND}_{\mathsf{MCFE},\mathcal{A}}(\lambda, n) \leq \mathsf{negl}(\lambda)$*.*

We omit n when it is clear from the context. We also often omit $\mathcal{A}$ from the parameter of experiments or games when it is clear from context.

2.3 Decentralized Multi-Client Functional Encryption

Now, we recap the definition of decentralized multi-client functional encryption (DMCFE) as introduced by Chotard et al. in [9]. As for our definition of MCFE, we separate the algorithm Setup which generates public parameters defining in particular the set of functions, from the algorithm KeyGen.

Definition 4 ***(Decentralized Multi-Client Functional Encryption)*** *Let* $\mathcal{F} = \{\mathcal{F}_\rho\}_\rho$ *be a family (indexed by* ρ*) of sets* $\mathcal{F}_\rho$ *of functions* $f\colon \mathcal{X}_{\rho,1} \times \cdots \times \mathcal{X}_{\rho,n_\rho} \to \mathcal{Y}_\rho$*. Let* $\mathsf{Labels} = \{0,1\}^*$ *or* $\{\bot\}$

be a set of labels. A decentralized multi-client functional encryption scheme (DMCFE) for the function family $\mathcal{F}$ and the label set Labels *is a tuple of six algorithms* DMCFE = (Setup, KeyGen, KeyDerShare, KeyDerComb, Enc, Dec)*:*

$\mathsf{Setup}(1^\lambda, 1^n)$ *is defined as for MCFE in Definition 2.*

$\mathsf{KeyGen}(\mathsf{pp})$**:** *Takes as input the public parameters* pp *and outputs n secret keys $\{\mathsf{sk_i}\}_{i\in[n]}$.*

$\mathsf{KeyDerShare}(\mathsf{pp}, \mathsf{sk}_i, f)$**:** *Takes as input the public parameters* pp*, a secret key sk_i from position i and a function $f \in \mathcal{F}_\rho$, and outputs a partial functional decryption key $\mathsf{sk}_{i,f}$.*

$\mathsf{KeyDerComb}(\mathsf{pp}, \mathsf{sk}_{1,f}, \ldots, \mathsf{sk}_{n,f})$**:** *Takes as input the public parameters* pp*, n partial functional decryption keys $\mathsf{sk}_{1,f}, \ldots, \mathsf{sk}_{n,f}$ and outputs the functional decryption key sk_f.*

$\mathsf{Enc}(\mathsf{pp}, \mathsf{sk}_i, x_i, \ell)$ *is defined as for MCFE in Definition 2.*

$\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}_f, \mathsf{ct}_{1,\ell}, \ldots, \mathsf{ct}_{n,\ell})$ *is defined as for MCFE in Definition 2.*

A scheme DMCFE *is correct, if for all $\lambda, n \in \mathbb{N}$, $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$, $f \in \mathcal{F}_\rho$, $\ell \in$ Labels, $x_i \in \mathcal{X}_{\rho,i}$, when $\{\mathsf{sk}_i\}_{i\in[n]} \leftarrow \mathsf{KeyGen}(\mathsf{pp})$, $\mathsf{sk}_{i,f} \leftarrow \mathsf{KeyDerShare}(\mathsf{sk}_i, f)$ for $i \in [n]$, and $\mathsf{sk}_f \leftarrow \mathsf{KeyDerComb}(\mathsf{pp}, \mathsf{sk}_{1,f}, \ldots, \mathsf{sk}_{n,f})$, we have*

$$\Pr\left[\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}_f, \mathsf{Enc}(\mathsf{pp}, \mathsf{sk}_1, x_1, \ell), \ldots, \mathsf{Enc}(\mathsf{pp}, \mathsf{sk}_n, x_n, \ell)) = f(x_1, \ldots, x_n)\right] = 1 \ .$$

We remark that there is no master secret key msk. Furthermore, similarly to [9], our definition does not explicitly ask the setup to be decentralized. However, all our constructions allow for the setup to be easily decentralized, at least assuming that the original schemes have such a property in the case of our compilers.

We consider a similar security definition for the decentralized multi-client scheme. We point out that contrary to [9], we do not differentiate encryption keys from secret keys. This is without loss of generality, as corruptions in [9] only allow to corrupt both keys at the same time.

Definition 5 ***(Security of DMCFE)*** *The* xx-yy*-IND security notion of an DMCFE scheme (*$\mathrm{xx} \in \{\mathrm{sta}, \mathrm{adt}\}$ *and* $\mathrm{yy} \in \{\mathrm{one}, \mathrm{any}, \mathrm{pos}\}$*) is similar to the one of an MCFE (Definition 3), except that there is no master secret key* msk *and the key derivation oracle is now defined as:*

Key derivation oracle $\mathsf{QKeyD}(f)$**:** *Computes $\mathsf{sk}_{i,f} := \mathsf{KeyDerShare}(\mathsf{pp}, \mathsf{sk}_i, f)$ for $i \in [n]$ and outputs $\{\mathsf{sk}_{i,f}\}_{i\in[n]}$.*

2.4 Inner-Product Functionality

We describe the functionalities supported by the constructions in this deliverable, by considering the index ρ of $\mathcal{F}$ in more detail.

The index of the family is defined as $\rho = (\mathcal{R}, n, m, X, Y)$ where $\mathcal{R}$ is either $\mathbb{Z}$ or $\mathbb{Z}_L$ for some integer L, and n, m, X, Y are positive integers. If X, Y are omitted, then $X = Y = L$ is used (i.e., no constraints).

This defines $\mathcal{F}_\rho = \{f_{\mathbf{y}_1,\ldots,\mathbf{y}_n} : (\mathcal{R}^m)^n \to \mathcal{R}\}$ where

$$f_{\mathbf{y}_1,\ldots,\mathbf{y}_n}(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \sum_{i=1}^{n} \langle \mathbf{x}_i, \mathbf{y}_i \rangle = \langle \mathbf{x}, \mathbf{y} \rangle \ ,$$

where the vectors satisfy the following bounds: $\|\mathbf{x}_i\|_\infty < X, \|\mathbf{y}_i\|_\infty < Y$ for $i \in [n]$, and where $\mathbf{x} \in \mathcal{R}^{mn}$ and $\mathbf{y} \in \mathcal{R}^{mn}$ are the vectors corresponding to the concatenation of the n vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and $\mathbf{y}_1, \ldots, \mathbf{y}_n$ respectively.

3 Decentralization

In this section, we describe the decentralization compiler by Abdalla et al. [2], developed in the context of the FENTEC project, which allows for the decentralization of MCFE schemes that satisfy an additional property, called special key derivation. We start by defining this property and describing instantiations of existing schemes [5, 9] that satisfy it. Next, we describe the compiler for the case in which the underlying modulus of the special key derivation property is prime. Finally, we extend it to the case where this modulus is a hard-to-factor composite number.

3.1 Special Key Derivation Property

Definition 6 (MCFE with Special Key Derivation) *An MCFE scheme* $\mathsf{MCFE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ *for a family of functions* $\mathcal{F}$ *and a set of labels* Labels *has the special key derivation property modulo* L *if:*[2]

- *Secret keys* sk_i *generated by* KeyGen *have the following form:* $\mathsf{sk}_i = (i, s_i, \{\mathbf{u}_i^k\}_{k\in[\kappa]})$*, where* $s_i \in \{0,1\}^*$*, and* $\mathbf{u}_i^k \in \mathbb{Z}_L^m$*, and* κ *and* m *are positive integers implicitly depending on the public parameters* pp*.*
- $\mathsf{sk}_f \leftarrow \mathsf{KeyDer}(\mathsf{pp}, \mathsf{msk}, f)$ *outputs* $\mathsf{sk}_f = (\{s_{i,f}\}_{i\in[n]}, \{\mathsf{dk}_f^k\}_{k\in[\kappa]})$*, where* $s_{i,f}$ *is a (polynomial-time) function of* pp*,* i*,* s_i*, and* f*, while:*

$$\mathsf{dk}_f^k = \sum_{i=1}^{n} \langle \mathbf{u}_i^k, \mathbf{y}_{i,f}^k \rangle = \langle \mathbf{u}^k, \mathbf{y}_f^k \rangle \ ,$$

  where $\mathbf{y}_{i,f}^k \in \mathbb{Z}_L^m$ *is a (polynomial-time) function of* pp*,* i*, and* f*, and* $\mathbf{u}^k$ *and* $\mathbf{y}_f^k$ *are the vectors in* $\mathbb{Z}_L^{mn}$ *corresponding to the concatenation of the vectors* $\{\mathbf{u}_i^k\}_{i\in[n]}$ *and* $\{\mathbf{y}_{i,f}^k\}_{i\in[n]}$ *respectively.*

Without loss of generality for MCFE with the special key derivation property, we can suppose that $\mathsf{msk} = \{\mathsf{sk}_i\}_{i\in[n]}$. We also remark that we do not require any property of the family of functions $\mathcal{F}$ and that our compiler could be applicable to more general MCFE than inner-product ones.

3.2 Instantiations

Abdalla et al. [2] observed that the following schemes fulfill the special key derivation property:

1. The MCFE construction by Chotard et al. [9, Section 4] satisfies the special key derivation property modulo $L = p$ (the order of the cyclic group), with $\kappa = 2$ and $\mathbf{y}_f^k = \mathbf{y}$, when $f : \mathbf{x} \mapsto \langle \mathbf{x}, \mathbf{y} \rangle$.
2. The generic constructions by Abdalla et al. [5, Section 3] (both over $\mathbb{Z}$ and $\mathbb{Z}_L$) satisfy the special key derivation property modulo L (where L is the modulo used for the information-theoretic MIFE/MCFE with one-time security) with $\mathbf{y}_f^k = \mathbf{y}$. The instantiations from MDDH and Paillier ([5, Section 4]) use $L = p$ the prime order of the cyclic group, and $L = N = pq$ the modulus used for Paillier, respectively. The instantiation from LWE ([5,

[2]The integer L can depend on the public parameters pp.

$\underline{\mathsf{Setup}'(1^\lambda, 1^n)}$:
Return $\mathsf{Setup}(1^\lambda, 1^n)$

$\underline{\mathsf{KeyGen}'(\mathsf{pp})}$:
$(\{\mathsf{sk}_i\}_{i\in[n]}, \mathsf{msk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$
Recall that $\mathsf{sk}_i = (i, s_i, \{\mathbf{u}_i^k\}_{k\in[\kappa]})$
For $k \in [\kappa]$:
For $i \in [n-1]$, $\mathbf{v}_i^k \leftarrow \mathbb{Z}_L^M$
$\mathbf{v}_n^k := -\sum_{i=1}^{n-1} \mathbf{v}_i^k \bmod L$
Return $\{\mathsf{sk}'_i = (\mathsf{sk}_i, \{\mathbf{v}_i^k\}_{k\in[\kappa]})\}_{i\in[n]}$

$\underline{\mathsf{Enc}'(\mathsf{pp}, \mathsf{sk}'_i, x_i, \ell)}$:
Parse $\mathsf{sk}'_i = (\mathsf{sk}_i, \{\mathbf{v}_i^k\}_{k\in[\kappa]})$
Return $\mathsf{ct}_{i,\ell} \leftarrow \mathsf{Enc}(\mathsf{pp}, \mathsf{sk}_i, x_i, \ell)$

$\underline{\mathsf{KeyDerShare}'(\mathsf{pp}, \mathsf{sk}'_i, f)}$:
Parse $\mathsf{sk}'_i = (\mathsf{sk}_i, \{\mathbf{v}_i^k\}_{k\in[\kappa]})$
For $k \in [\kappa]$, $\mathsf{dk}_{i,f}^k := \langle \mathbf{u}_i^k, \mathbf{y}_{i,f}^k \rangle + \langle \mathbf{v}_i^k, \mathbf{y}_f^k \rangle$
Return $\mathsf{sk}'_{i,f} := (s_{i,f}, \{\mathsf{dk}_{i,f}^k\}_{k\in[\kappa]})$

$\underline{\mathsf{KeyDerComb}'(\mathsf{pp}, \{\mathsf{sk}'_{i,f}\}_{i\in[n]})}$:
Parse $\{\mathsf{sk}'_{i,f} = (s_{i,f}, \{\mathsf{dk}_{i,f}^k\}_{k\in[\kappa]})\}_{i\in[n]}$
For $k \in [\kappa]$, $\mathsf{dk}_f^k := \sum_{i=1}^{n} \mathsf{dk}_{i,f}^k$
Return $\mathsf{sk}'_f = (\{s_{i,f}\}_{i\in[n]}, \{\mathsf{dk}_f^k\}_{k\in[\kappa]})$

$\underline{\mathsf{Dec}'(\mathsf{pp}, \mathsf{sk}'_f, \{\mathsf{ct}_{i,\ell}\}_{i\in[n]})}$:
Return $\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}'_f, \{\mathsf{ct}_{i,\ell}\}_{i\in[n]})$

Figure 2: Compiler from MCFE to DMCFE′: $s_{i,f}$ is a function of pp, i, s_i, f and $\mathbf{y}_{i,f}^k$ is a function of pp, i, f, and k. $M = mn$.

Section 4]) requires L to be a large enough prime number so that computations modulo L and over the integers are the same with overwhelming probability over the generation of msk.

3.3 Compiler for Prime Moduli

We start by presenting the compiler from MCFE schemes with the special key derivation property modulo a prime L by Abdalla et al. [2] in Fig. 2. The correctness of this scheme, as shown in [2], follows directly from the fact that:

$$\sum_{i=1}^{n} \mathsf{dk}_{i,f}^k = \sum_{i=1}^{n} \langle \mathbf{u}_i^k, \mathbf{y}_{i,f}^k \rangle + \sum_{i=1}^{n} \langle \mathbf{v}_i^k, \mathbf{y}_f^k \rangle$$
$$= \mathsf{dk}_f^k + \langle \sum_{i=1}^{n} \mathbf{v}_i^k, \mathbf{y}_f^k \rangle = \mathsf{dk}_f^k + \langle \mathbf{0}, \mathbf{y}_f^k \rangle = \mathsf{dk}_f^k \ .$$

Note that, while the vectors $\mathbf{u}_i^k$ and $\mathbf{y}_{i,f}^k$ are m-dimensional, vectors $\mathbf{v}_i^k$ and $\mathbf{y}_f^k$ are (mn)-dimensional.

The security theorem of the compiler, for which the proof can be found in [2], is stated as follows.

Theorem 1 *Let* MCFE = (Setup, KeyGen, KeyDer, Enc, Dec) *be an MCFE construction for a family of functions* $\mathcal{F}$ *and a set of labels* Labels. *We suppose that* MCFE *has the special key derivation property modulo a prime* L. *For any* xx $\in$ {sta, adt} *and any* yy $\in$ {one, pos, any}, *if* MCFE *is an* xx-yy*-IND-secure MCFE scheme, then the scheme* DMCFE′ *depicted in Fig. 2 is an* xx-yy*-IND-secure DMCFE scheme. Namely, for any PPT adversary* $\mathcal{A}$, *there exist a PPT adversary* $\mathcal{B}$ *such*

that:

$$\mathsf{Adv}^{\text{xx-yy-IND}}_{\mathsf{DMCFE}',\mathcal{A}}(\lambda, n) \leq \mathsf{Adv}^{\text{xx-yy-IND}}_{\mathsf{MCFE},\mathcal{B}}(\lambda, n) \ .$$

3.4 Extension to Hard-to-Factor Moduli

Abdalla et al. extended the previous scheme to moduli L which are hard to factor. This is required for the Paillier instantiation from [5, Section 4.3].

We recap the formal details presented in [2] here.

Definition 7 (Factorization) *Let* GenL *be a PPT algorithm taking as input the security parameter* 1^λ *and outputing a number* $L \geq 2$*. We define the experiment* $\mathsf{Factor}_{\mathsf{GenL}}(\lambda, \mathcal{A})$ *for an adversary* $\mathcal{A}$ *as follows: it outputs* 1 *if on input* $L \leftarrow \mathsf{GenL}(1^\lambda)$*, the adversary outputs two integers* $L_1, L_2 \geq 2$*, such that* $L_1 \cdot L_2 = L$*. The advantage of* $\mathcal{A}$ *is* $\mathsf{Adv}^{\mathsf{Factor}}_{\mathsf{GenL},\mathcal{A}}(\lambda) = \Pr[\mathsf{Factor}_{\mathsf{GenL}}(\lambda, \mathcal{A})]$*. Factorization is hard for* GenL *if the advantage of any PPT adversary* $\mathcal{A}$ *is negligible in* λ*.*

Now, we recap the security theorem for the decentralization compiler for hard-to-factor moduli of [2].

Theorem 2 *Let* $\mathsf{MCFE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ *be an MCFE construction for an ensemble of functions* $\mathcal{F}$ *and a set of labels* Labels*. We suppose that* MCFE *has the special key derivation property modulo an integer* L*, which is part of the public parameter* pp *and generated as* $L \leftarrow \mathsf{GenL}(1^\lambda)$ *in the setup, for some polynomial-time algorithm. We assume that factorization is hard for* GenL*. For any* $\mathrm{xx} \in \{\mathrm{sta}, \mathrm{adt}\}$ *and any* $\mathrm{yy} \in \{\mathrm{one}, \mathrm{pos}, \mathrm{any}\}$*, if* MCFE *is an* xx-yy*-IND-secure MCFE scheme, then the scheme* DMCFE' *depicted in Fig. 2 is an* xx-yy*-IND-secure DMCFE scheme. Namely, for any PPT adversary* $\mathcal{A}$*, there exist two PPT adversaries* $\mathcal{B}$ *and* $\mathcal{B}'$ *such that:*

$$\mathsf{Adv}^{\text{xx-yy-IND}}_{\mathsf{DMCFE}',\mathcal{A}}(\lambda, n) \leq \mathsf{Adv}^{\text{xx-yy-IND}}_{\mathsf{MCFE},\mathcal{B}}(\lambda, n) + 2 \cdot \mathsf{Adv}^{\mathsf{Factor}}_{\mathsf{GenL},\mathcal{B}'}(\lambda) \ .$$

The proof for this theorem can be found in [2].

4 Security of the MCFE from Abdalla et al. against Adaptive Corruptions

In this section, we recap the observation by Abdalla et al. in [2] that shows that the MIFE scheme by Abdalla et al. [5] is also secure against adaptive corruptions, when their unique encryption and secret key is split into individual secret keys for each party in a natural way, and therefore fulfills the notion of MCFE. The splitting of the master secret keys into individual keys is described in Fig. 3 and Fig. 4.

The transformation presented here only allows the construction of a multi-client functional encryption scheme without labels. In D4.10, we present a recent result by Abdalla et al. [1], which shows how to turn a single-input functional encryption scheme into a multi-client functional encryption scheme with labels.

For simplicity, as in [2], we focus here on the bounded-norm MCFE case since the construction over $\mathbb{Z}_L$ can be easily adapted from it. Towards this goal, Section 4.1 first recalls the definition of FE with two-step decryption and linear encryption as introduced in [5]. Next, Section 4.2 recalls the other building block, an *sta-one-IND-secure* MCFE scheme for $\mathcal{F}_\rho, \rho = (\mathbb{Z}_L, n, m, L, L)$. Finally, Section 4.3 recalls the MCFE construction from [5].

4.1 Inner-Product FE with Two-Step Decryption and Linear Encryption

The construction by Abdalla et al. [5] extends a one-time secure MCFE scheme over $\mathbb{Z}_L$ to a many-time secure MCFE scheme over $\mathbb{Z}$. This extension relies on a single-input FE scheme for $\mathcal{F}_\rho, \rho = (\mathbb{Z}, 1, m, X, Y)$ satisfying two properties, called *two-step decryption* and *linear encryption* [5]. As indicated in [5], the *two-step decryption* property informally says that the FE decryption algorithm can be broken in two steps: one step that uses the secret key to return an encoding of the result and the other step that returns the actual result $\langle \mathbf{x}, \mathbf{y} \rangle$ as long as the bounds $||\mathbf{x}||_\infty < X, ||\mathbf{y}||_\infty < Y$ hold. The *linear encryption* property, on the other hand, informally states that the FE encryption algorithm is additively homomorphic. We now recall these definitions more formally.

Definition 8 (Two-step decryption [5]) *A secret-key FE scheme* $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ *for the function ensemble* $\mathcal{F}_\rho, \rho = (\mathbb{Z}, 1, m, X, Y)$ *satisfies the* two-step decryption *property if it admits PPT algorithms* $\mathsf{Setup}^\star$, $\mathsf{Dec}_1, \mathsf{Dec}_2$ *and an encoding function* $\mathcal{E}$ *such that:*

1. *For all* $\lambda \in \mathbb{N}, \mathsf{Setup}^\star(1^\lambda, 1^n)$ *outputs* pp *where* pp *includes* $\rho = (\mathbb{Z}, 1, m, X, Y)$ *and a bound* $B \in \mathbb{N}$*, as well as the description of a group* $\mathbb{G}$ *(with group law* $\circ$*) of order* $L > 2 \cdot n \cdot m \cdot X \cdot Y$*, which defines the encoding function* $\mathcal{E} : \mathbb{Z}_L \times \mathbb{Z} \to \mathbb{G}$*.*

2. *For all* $\mathsf{msk} \leftarrow \mathsf{KeyGen}(\mathsf{pp}), \mathbf{x} \in \mathbb{Z}^m, \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pp}, \mathsf{msk}, \mathbf{x}), \mathbf{y} \in \mathbb{Z}^m$*, and* $\mathsf{sk} \leftarrow \mathsf{KeyDer}(\mathsf{msk}, \mathbf{y})$*, we have*

$$\mathsf{Dec}_1(\mathsf{pp}, \mathsf{sk}, \mathsf{ct}) = \mathcal{E}(\langle \mathbf{x}, \mathbf{y} \rangle \bmod L, \mathsf{noise}) \ ,$$

for some $\mathsf{noise} \in \mathbb{N}$ *that depends on* ct *and* sk*. Furthermore, it holds that* $Pr[\mathsf{noise} < B] = 1 - \mathsf{negl}(\lambda)$*, where the probability is taken over the random coins of* KeyGen *and* KeyDer*. Note that there is no restriction on the norm of* $\langle \mathbf{x}, \mathbf{y} \rangle$ *here.*

Define $\mathsf{pp}_{\mathsf{ot}} = (n, m, L)$

$\underline{\mathsf{KeyGen}^{\mathsf{ot}}(\mathsf{pp}_{\mathsf{ot}}) :}$
$\{\mathbf{u}_i\}_{i\in[n]} \leftarrow (\mathbb{Z}_L^m)^n$
Return $\mathsf{msk} := \{\mathsf{msk}_i\}_{i\in[n]} = \{\mathbf{u}_i\}_{i\in[n]}$

$\underline{\mathsf{Enc}^{\mathsf{ot}}(\mathsf{pp}_{\mathsf{ot}}, \mathsf{msk}_i, \mathbf{x}_i) :}$
Parse $\mathsf{msk}_i = \mathbf{u}_i$
Return $\mathsf{ct}_i := \mathbf{u}_i + \mathbf{x}_i \bmod L$

$\underline{\mathsf{KeyDer}^{\mathsf{ot}}(\mathsf{pp}_{\mathsf{ot}}, \mathsf{msk}, \mathbf{y}) :}$
Parse $\mathsf{msk} = \{\mathbf{u}_i\}_{i\in[n]}, \mathbf{y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$
Return $\mathsf{dk}_{\mathbf{y}} := \sum_{i\in[n]} \langle \mathbf{u}_i, \mathbf{y}_i \rangle$

$\underline{\mathsf{Dec}^{\mathsf{ot}}(\mathsf{pp}_{\mathsf{ot}}, \mathsf{dk}_{\mathbf{y}}, \mathbf{y}, \{\mathsf{ct}_i\}_{i\in[n]}) :}$
Parse $\mathbf{y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$
Return $\sum_{i\in[n]} \langle \mathsf{ct}_i, \mathbf{y}_i \rangle - \mathsf{dk}_{\mathbf{y}} \bmod L$

Figure 3: One-Time Inner-Product MCFE over $\mathbb{Z}_L$ (for $\mathcal{F}_{L,n}^m$)

3. *Given any* $\gamma \in \mathbb{Z}_L$, *and* pp, *one can efficiently compute* $\mathcal{E}(\gamma, 0)$.
4. *The encoding* $\mathcal{E}$ *is linear, that is: for all* $\gamma, \gamma' \in \mathbb{Z}_L, \mathsf{noise}, \mathsf{noise}' \in \mathbb{Z}$, *we have*
$$\mathcal{E}(\gamma, \mathsf{noise}) \circ \mathcal{E}(\gamma', \mathsf{noise}') = \mathcal{E}(\gamma + \gamma' \bmod L, \mathsf{noise} + \mathsf{noise}') \ .$$
5. *For all* $\gamma < 2 \cdot n \cdot m \cdot X \cdot Y$, *and* $\mathsf{noise} < n \cdot B, \mathsf{Dec}_2(\mathsf{pp}, \mathcal{E}(\gamma, \mathsf{noise})) = \gamma$.

Definition 9 (Linear encryption [5]) *A secret-key FE scheme* $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ *is said to satisfy the linear encryption property if there exists a deterministic algorithm* Add *that takes as input a ciphertext and a message, such that for all* $\mathbf{x}, \mathbf{x}' \in \mathbb{Z}^m$, *the following are identically distributed:*
$$\mathsf{Add}(\mathsf{Enc}(\mathsf{pp}, \mathsf{msk}, \mathbf{x}), \mathbf{x}'), \quad \textit{and} \quad \mathsf{Enc}\big(\mathsf{pp}, \mathsf{msk}, (\mathbf{x} + \mathbf{x}' \bmod L)\big) \ .$$
Recall that the value $L \in \mathbb{N}$ *is defined as part of the output of the algorithm* $\mathsf{Setup}^\star$ *(see the two-step decryption property above).*

4.2 One-Time Inner-Product MCFE over $\mathbb{Z}_L$

We recap the one-time secure scheme provided by Abdalla et al. [5] in Fig. 3. The modifications made in [2] are the following:

1. The description does not contain a setup procedure $\mathsf{Setup}^{\mathsf{ot}}$, which now simply defines (n, m, L).
2. The steps of the original $\mathsf{Setup}^{\mathsf{ot}}$ in [5] are now defined in the $\mathsf{KeyGen}^{\mathsf{ot}}$ procedure. Also the unique secret key is split into individual secret keys for each party.

Since these modifications do not impact the correctness of the scheme, we refer to [5] for a proof of correctness. As for its security with respect to adaptive corruptions, we need to modify the proof by Abdalla et al. [5] to account for corruption queries.

Theorem 3 *The* $\mathsf{MCFE}^{\mathsf{ot}}$ *scheme in Figure 3 is* adt-one-IND *secure. Namely, for any adversary* $\mathcal{A}$, $\mathsf{Adv}^{\text{adt-one-IND}}_{\mathsf{MCFE}^{\mathsf{ot}},\mathcal{A}}(\lambda) = 0$

The proof of this theorem can be found in [2].

$\underline{\mathsf{Setup}(1^\lambda, 1^n)}$:

$\mathsf{pp}_{\mathsf{si}} \leftarrow \mathsf{Setup}^{\mathsf{si}}(1^\lambda, 1^n)$

Set $\mathsf{pp}_{\mathsf{ot}} := (n, m, L)$, with $\rho_{\mathsf{si}} = (\mathbb{Z}, 1, m, 3X, Y)$ and L implicitly defined from $\mathsf{pp}_{\mathsf{si}}$

Return $\mathsf{pp} = (\mathsf{pp}_{\mathsf{si}}, \mathsf{pp}_{\mathsf{ot}})$

$\underline{\mathsf{KeyGen}(\mathsf{pp})}$:

$\{\mathbf{u}_i\}_{i\in[n]} \leftarrow \mathsf{KeyGen}^{\mathsf{ot}}(\mathsf{pp}_{\mathsf{ot}})$

For $i \in [n]$, $\mathsf{msk}_i^{\mathsf{si}} \leftarrow \mathsf{KeyGen}^{\mathsf{si}}(\mathsf{pp}_{\mathsf{si}})$, $\mathsf{sk}_i := (\mathsf{msk}_i^{\mathsf{si}}, \mathbf{u}_i)$

Return $\{\mathsf{sk}_i\}_{i\in[n]}$

$\underline{\mathsf{Enc}(\mathsf{pp}, \mathsf{sk}_i, \mathbf{x}_i)}$:

Parse $\mathsf{sk}_i = (\mathsf{msk}_i^{\mathsf{si}}, \mathbf{u}_i)$ and return $\mathsf{ct}_i := \mathsf{Enc}^{\mathsf{si}}(\mathsf{pp}_{\mathsf{si}}, \mathsf{msk}_i^{\mathsf{si}}, \mathsf{Enc}^{\mathsf{ot}}(\mathsf{pp}_{\mathsf{ot}}, \mathbf{u}_i, \mathbf{x}_i))$

$\underline{\mathsf{KeyDer}(\mathsf{pp}, \mathsf{msk}, \mathbf{y})}$:

Parse $\mathsf{msk} = \{\mathsf{msk}_i^{\mathsf{si}}, \mathbf{u}_i\}_{i\in[n]}, \mathbf{y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$

For $i \in [n]$, $\mathsf{sk}_{i,\mathbf{y}} \leftarrow \mathsf{KeyDer}^{\mathsf{si}}(\mathsf{pp}_{\mathsf{si}}, \mathsf{msk}_i^{\mathsf{si}}, \mathbf{y}_i)$

$\mathsf{dk}_{\mathbf{y}} := \mathsf{KeyDer}^{\mathsf{ot}}(\mathsf{pp}_{\mathsf{ot}}, \{\mathbf{u}_i\}_{i\in[n]}, \mathbf{y})$

Return $\mathsf{sk}_{\mathbf{y}} := (\{\mathsf{sk}_{i,\mathbf{y}}\}_{i\in[n]}, \mathsf{dk}_{\mathbf{y}})$

$\underline{\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}_{\mathbf{y}}, \{\mathsf{ct}_i\}_{i\in[n]})}$:

Parse $\mathsf{sk}_{\mathbf{y}} = (\{\mathsf{sk}_{i,\mathbf{y}}\}_{i\in[n]}, \mathsf{dk}_{\mathbf{y}})$

For $i \in [n]$, $\mathcal{E}(\langle \mathbf{u}_i + \mathbf{x}_i, \mathbf{y}_i\rangle \bmod L, \mathsf{noise}_i) \leftarrow \mathsf{Dec}_1^{\mathsf{si}}(\mathsf{pp}_{\mathsf{si}}, \mathsf{sk}_{i,\mathbf{y}}, \mathsf{ct}_i)$

Return $\mathsf{Dec}_2^{\mathsf{si}}(\mathsf{pp}_{\mathsf{si}}, \mathcal{E}(\langle \mathbf{u}_1 + \mathbf{x}_1, \mathbf{y}_1\rangle \bmod L, \mathsf{noise}_1)) \circ \ldots$
$\circ\, \mathcal{E}(\langle \mathbf{u}_n + \mathbf{x}_n, \mathbf{y}_n\rangle \bmod L, \mathsf{noise}_n) \circ \mathcal{E}(-\mathsf{dk}_{\mathbf{y}}, 0))$

Figure 4: Inner-Product for $\mathcal{F}_\rho, \rho = (\mathbb{Z}, n, m, X, Y)$ built from $\mathsf{MCFE}^{\mathsf{ot}}$ for $\mathcal{F}_{\rho_{\mathsf{ot}}}, \rho_{\mathsf{ot}} = (\mathbb{Z}_L, n, m, L, L)$ and FE for $\mathcal{F}_{\rho_{\mathsf{si}}}, \rho_{\mathsf{si}} = (\mathbb{Z}, 1, m, 3X, Y)$

4.3 Inner-Product MCFE over $\mathbb{Z}$

We recall the construction by Abdalla et al. [5] of a pos-IND-secure scheme $\mathsf{MCFE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ from the (one-IND-secure) MCFE scheme $\mathsf{MCFE}^{\mathsf{ot}} = (\mathsf{KeyGen}^{\mathsf{ot}}, \mathsf{KeyDer}^{\mathsf{ot}}, \mathsf{Enc}^{\mathsf{ot}}, \mathsf{Dec}^{\mathsf{ot}})$ (described in Section 4.2) and from any single-input any-IND-secure scheme $\mathsf{FE} = (\mathsf{Setup}^{\mathsf{si}}, \mathsf{KeyGen}^{\mathsf{si}}, \mathsf{KeyDer}^{\mathsf{si}}, \mathsf{Enc}^{\mathsf{si}}, \mathsf{Dec}^{\mathsf{si}})$, in Fig. 4.

As for the one-time scheme in Section 4.2, we also present the modifications made in [2] for the adaptation ot the multi-client setting. The main modification is the modified KeyGen procedure, in order to split the unique secret key into individual secret keys for each party. Since this modification does not impact the correctness of the scheme, we refer to [5] for the proof of this property.

The security theorem for this construction is as follows.

Theorem 4 *Assume that the single-input scheme* FE *is any-IND-secure and that the multi-client scheme* $\mathsf{MCFE}^{\mathsf{ot}}$ *is adt-one-IND-secure. Then the multi-client scheme* MCFE *is adt-pos-IND-*

secure. Namely, for any PPT adversary $\mathcal{A}$, there exist PPT adversaries $\mathcal{B}$ and $\mathcal{B}'$ such that

$$\mathsf{Adv}^{\text{adt-pos-IND}}_{\mathsf{MCFE},\mathcal{A}}(\lambda, n) \leq \mathsf{Adv}^{\text{adt-one-IND}}_{\mathsf{MCFE}^{\text{ot}},\mathcal{B}}(\lambda, n) + n \cdot \mathsf{Adv}^{\text{any-IND}}_{\mathsf{FE},\mathcal{B}'}(\lambda, n) \ .$$

The security proof for this theorem can be found in [2]

5 LWE Instantiations

In this section, we recall the description of two LWE-based (single-input) FE schemes which can be used to instantiate the MCFE schemes in Section 4. The first one is by Agrawal et al. [6, Section 4.1] and the second one is by Abdalla et al. [4].

5.1 Inner-product functional encryption from [6]

The many-AD-IND secure Inner-Product FE by Agrawal et al. [6, Section 4.1] is recalled in Fig. 5. The proof that it satisfies the two-step decryption and the linear encryption properties can be found in [5].

$\underline{\mathsf{Setup}(1^\lambda, \mathcal{F}_1^{m,X,Y})}$:
Let $N = N(\lambda)$, and set integers $M, q \geq 2$, real $\alpha \in (0,1)$, and distribution D over $\mathbb{Z}^{m\times M}$ as explained in [6]; set $K := m \cdot X \cdot Y$, $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{M\times N}$, $\mathbf{Z} \xleftarrow{R} \mathsf{D}$, $\mathbf{U} := \mathbf{ZA} \in \mathbb{Z}_q^{m\times N}$, $\mathsf{pk} := (K, \mathbf{A}, \mathbf{U})$, $\mathsf{msk} := \mathbf{Z}$.
Return $(\mathsf{pk}, \mathsf{msk})$

$\underline{\mathsf{Enc}(\mathsf{pk}, \mathbf{x} \in \mathbb{Z}^m)}$:
$\mathbf{s} \xleftarrow{R} \mathbb{Z}_q^N$, $\mathbf{e}_0 \xleftarrow{R} D_{\mathbb{Z},\alpha q}^M$, $\mathbf{e}_1 \xleftarrow{R} D_{\mathbb{Z},\alpha q}^m$
$\mathbf{c}_0 := \mathbf{As} + \mathbf{e}_0 \in \mathbb{Z}_q^M$
$\mathbf{c}_1 := \mathbf{Us} + \mathbf{e}_1 + \mathbf{x} \cdot \lfloor \frac{q}{K} \rfloor \in \mathbb{Z}_q^m$
Return $\mathsf{ct}_\mathbf{x} := (\mathbf{c}_0, \mathbf{c}_1)$

$\underline{\mathsf{KeyGen}(\mathsf{msk}, \mathbf{y} \in \mathbb{Z}^m)}$:
Return $\mathsf{sk}_\mathbf{y} := \begin{pmatrix} \mathbf{Z}^\top \mathbf{y} \\ \mathbf{y} \end{pmatrix} \in \mathbb{Z}^{M+m}$

$\underline{\mathsf{Dec}(\mathsf{sk}_\mathbf{y}, \mathsf{ct}_\mathbf{x})}$:
$\mu' := \begin{pmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \end{pmatrix}^\top \mathsf{sk}_\mathbf{y} \mod q.$
Return $\mu \in \{-K+1, \ldots, K-1\}$ that minimizes $\left| \lfloor \frac{q}{K} \rfloor \mu - \mu' \right|$.

Figure 5: Functional encryption scheme by Agrawal et al. [6] for the class $\mathcal{F}_1^{m,X,Y}$ based on the LWE assumption.

5.2 Inner-product functional encryption from [4]

The inner-product FE scheme by Abdalla, Bourse, De Caro, and Pointcheval in [4] is an extension of inner-product FE construction in [3]. It achieves adaptive security and has instantiations based on the ElGamal (plain DDH assumption) [10], Paillier/Bresson-Chevassut-Pointcheval (DCR assumption) [8], and Regev (LWE assumption) [13] encryption schemes. For simplicity, we adopt here the same notation from [4] in the description of the scheme.

Fig. 6 describes the instantiation based on the Regev encryption scheme [13]. The proof that it

satisfies the two-step decryption and linear encryption properties is similar to the one given in [5] for the [6] FE scheme.

$\underline{\mathsf{Setup}(1^\lambda, \mathcal{F}_1^{m,X,Y})}$:
Let n, m, p, q be integer parameters
Let σ a positive real parameter such that they verify the conditions required
Let $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{m \times n}$ be a uniformly random matrix.
Set $\mathsf{pp} = (\lambda, \ell, n, m, p, q, \mathbf{A})$
Sample $(\mathbf{s}_0, \mathbf{e}_0) \xleftarrow{R} \mathbb{Z}_q^n \times \chi_{\gamma_0}^m$
Sample $\mathbf{b}_0 \leftarrow \mathbf{A}\mathbf{s}_0 + \mathbf{e}_0 \in \mathbb{Z}_q^m$
For all $i \in [\ell]$, set $(t_i, \mathbf{s}_i, \mathbf{e}_i) \xleftarrow{R} \{0, \ldots, T\} \times \mathbb{Z}_q^n \times \chi_\sigma^m$
For all $i \in [\ell]$, set $\mathbf{b}_i \leftarrow \mathbf{A}(t_i \cdot \mathbf{s}_0 + \mathbf{s}_i) + \mathbf{e}_i \in \mathbb{Z}_q^m$
$\mathsf{msk} = (\mathbf{s}_i, t_i)_{i \in [\ell]}$
$\mathsf{pk} = (\mathbf{b}_0, \mathbf{b}_i)_{i \in [\ell]}$
Return $(\mathsf{pk}, \mathsf{msk})$

$\underline{\mathsf{Enc}(\mathsf{pk}, \mathbf{x} \in \mathcal{M}_x)}$:
Pick $\mathbf{r} \xleftarrow{R} \{0,1\}^m$
Set $\mathsf{ct}_0 \leftarrow \mathbf{A}^\intercal \mathbf{r} \in \mathbb{Z}_q^n$
Set $\mathsf{ct}_1 \leftarrow \mathbf{b}_0^\intercal \mathbf{r} \in \mathbb{Z}_q$
For all $i \in [\ell]$, $\mathsf{ct}_{2,i} \leftarrow \mathbf{b}_i^\intercal \mathbf{r} + t(x_i) \in \mathbb{Z}_q$, where $t(v) = v \cdot \lfloor q/p \rfloor \in \mathbb{Z}_q$.
Return $\mathsf{ct}_\mathbf{x} = (\mathsf{ct}_0, \mathsf{ct}_1, (\mathsf{ct}_{2,i})_{i \in [\ell]})$

$\underline{\mathsf{KeyGen}(\mathsf{msk}, \mathbf{y} \in \mathcal{M}_y)}$:
Set $\mathbf{s}_\mathbf{y} \leftarrow \sum_{i \in [\ell]} y_i \mathbf{s}_i \in \mathbb{Z}_q^n$
Set $t_\mathbf{y} \leftarrow \sum_{i \in [\ell]} y_i t_i \in \mathbb{Z}$
Return $\mathsf{sk}_\mathbf{y} = (\mathbf{s}_\mathbf{y}, t_\mathbf{y})$

$\underline{\mathsf{Dec}(\mathsf{sk}_\mathbf{y}, \mathsf{ct}_\mathbf{x})}$:
Set $\mathsf{ct}_{\langle \mathbf{x}, \mathbf{y} \rangle} \leftarrow \sum_{i \in [\ell]} y_i \mathsf{ct}_{2,i} - t_\mathbf{y} \mathsf{ct}_1 - \mathsf{ct}_0^\intercal \mathsf{sk}_\mathbf{y} \in \mathbb{Z}_q$.
Return the plaintext m, where m is such that $d - t(m) \in \mathbb{Z}_q$ is closest to $0 \mod q$.

Figure 6: Functional encryption scheme by Abdalla et al. [4] for the class $\mathcal{F}_1^{m,X,Y}$ based on the LWE assumption.

According to [4], the message space is $\mathcal{M}_x = \{0, \ldots, M_x\} \subseteq \mathbb{Z}_p$ for some integer M_x and prime $p > \ell M_x M_y$. $\mathcal{T} = \{0, \ldots, T\}^\ell$, where T is set according to the security properties needed. T/M_x super-polynomial is needed for security against polynomially bounded adversaries, T/M_x exponential provides security against sub-exponentially bounded adversaries, where M_x is the biggest possible coordinate of any vector in $\mathcal{M}_x$.

In order for the proof of security to carry through, as well as the correctness, the following properties on the parameters have to be verified:

1. $m \geq (n + \ell + 2) \log q + 2 \log \frac{1}{\epsilon} + \Omega(1)$;
2. $T = M_x \cdot \lambda^{\omega(1)}$;
3. $\sigma \geq (1 + T\sqrt{\ell})\sigma'$;

4. $\gamma_0 > \sqrt{\frac{\ln(2\ell(1+1/\epsilon))}{\pi}}$;
5. $\sigma' q > 2\sqrt{n}$;
6. $p > \ell M_x M_y$;
7. $\frac{q}{2p} > \sigma M_y^2 \ell \sqrt{2m\lambda}$.

6 Conclusion

In this document, we extended the specifications of quantum-safe functional encryption as initially stated in Deliverable 4.7 in the context of the FENTEC project. In particular, we showed how to decentralize existing functional encryption schemes by using information theoretic techniques. This solves the problem mentioned in Deliverable 4.1 and 4.7, about the suitability of certain applications under a central authority for the functional key generation. This result is described in Section 3. In addition, we also presented an extension to the multi-client setting of the MIFE construction for the inner-product functionality presented in D4.7. This is described in Section 4. Finally, as in Deliverable 4.7, we recapped the quantum-safe schemes in Section 5, which can be used to instantiate the presented constructions in this deliverable.

References

[1] Michel Abdalla, Fabrice Benhamouda, and Romain Gay. From single-input to multi-client inner-product functional encryption. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019, Part III*, volume 11923 of *Lecture Notes in Computer Science*, pages 552–582, Kobe, Japan, December 8–12, 2019. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-34618-8_19. (Page 11.)

[2] Michel Abdalla, Fabrice Benhamouda, Markulf Kohlweiss, and Hendrik Waldner. Decentralizing inner-product functional encryption. In Dongdai Lin and Kazue Sako, editors, *PKC 2019: 22nd International Conference on Theory and Practice of Public Key Cryptography, Part II*, volume 11443 of *Lecture Notes in Computer Science*, pages 128–157, Beijing, China, April 14–17, 2019. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-17259-6_5. (Pages v, 1, 2, 8, 9, 10, 11, 12, 13, and 14.)

[3] Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Simple functional encryption schemes for inner products. In Jonathan Katz, editor, *PKC 2015: 18th International Conference on Theory and Practice of Public Key Cryptography*, volume 9020 of *Lecture Notes in Computer Science*, pages 733–751, Gaithersburg, MD, USA, March 30 – April 1, 2015. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-46447-2_33. (Page 15.)

[4] Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Better security for functional encryption for inner product evaluations. Cryptology ePrint Archive, Report 2016/011, 2016. http://eprint.iacr.org/2016/011. (Pages ii, iii, 1, 2, 15, and 16.)

[5] Michel Abdalla, Dario Catalano, Dario Fiore, Romain Gay, and Bogdan Ursu. Multi-input functional encryption for inner products: Function-hiding realizations and constructions without pairings. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 597–627, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-96884-1_20. (Pages v, 1, 2, 8, 9, 10, 11, 12, 13, 15, and 16.)

[6] Shweta Agrawal, Benoît Libert, and Damien Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part III*, volume 9816 of *Lecture Notes in Computer Science*, pages 333–362, Santa Barbara, CA, USA, August 14–18, 2016. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-53015-3_12. (Pages ii, iii, 1, 2, 15, and 16.)

[7] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011: 8th Theory of Cryptography Conference*, volume 6597 of *Lecture Notes in Computer Science*, pages 253–273, Providence, RI, USA, March 28–30, 2011. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-19571-6_16. (Page 1.)

[8] Emmanuel Bresson, Dario Catalano, and David Pointcheval. A simple public-key cryptosystem with a double trapdoor decryption mechanism and its applications. In Chi-Sung Laih, editor, *Advances in Cryptology – ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 37–54, Taipei, Taiwan, November 30 – December 4, 2003. Springer, Heidelberg, Germany. doi:10.1007/978-3-540-40061-5_3. (Page 15.)

[9] Jérémy Chotard, Edouard Dufour Sans, Romain Gay, Duong Hieu Phan, and David Pointcheval. Decentralized multi-client functional encryption for inner product. In Thomas Peyrin and Steven Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018, Part II*, volume 11273 of *Lecture Notes in Computer Science*, pages 703–732, Brisbane, Queensland, Australia, December 2–6, 2018. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-03329-3_24. (Pages 1, 4, 5, 6, and 8.)

[10] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985. (Page 15.)

[11] Shafi Goldwasser, S. Dov Gordon, Vipul Goyal, Abhishek Jain, Jonathan Katz, Feng-Hao Liu, Amit Sahai, Elaine Shi, and Hong-Sheng Zhou. Multi-input functional encryption. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 578–602, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-55220-5_32. (Pages 1 and 4.)

[12] Adam O'Neill. Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556, 2010. http://eprint.iacr.org/2010/556. (Page 1.)

[13] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th Annual ACM Symposium on Theory of Computing*, pages 84–93, Baltimore, MA, USA, May 22–24, 2005. ACM Press. doi:10.1145/1060590.1060603. (Page 15.)

[14] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997. (Page v.)